平成 14 年度

学士学位論文

モバイルコンテンツ課金プロトコル

A Billing Protocol for Mobile Communications

1030262　　岸田 光生

指導教員　　清水 明宏

2003 年 2 月 12 日

高知工科大学 情報システム工学科

要 旨

モバイルコンテンツ課金プロトコル

岸田 光生

近年，モバイル端末の急速な普及により，携帯電話などからのインターネット接続率が増加している．それに伴いコンテンツでは音楽データや画像データなど多数のデータ交換が盛んに行われている．データ交換を円滑に進めるために，料金支払いの問題が挙げられる．そこで課金システムの確立が必要となってくる．現在，課金システムとして料金代行サービスが主に行なわれている．しかし，公式サイト内での課金しか対応できていないため，非公式サイトでの課金システムが確立されていない．そこで，モバイル端末でも計算処理可能なSAS 認証方式を使ったSAS-Coin を利用する．しかし，SAS-Coin は，モバイル端末上での計算処理が可能であっても，計算量が増えると処理が重くなる．本研究では処理を軽くするため，SAS-2 認証方式を使った新しいSAS-Coin を検討し，安全かつ素早い方式を確立させ，小額課金システムの構築を行う．

キーワード　SAS，SAS-Coin，モバイル，課金プロトコル

Abstract

A Billing Protocol for Mobile Communications

Mitsuo Kishida

In recent years, mobile terminals are spreading, and mobile communications are increasing. Then music, image, and movie data are exchanged on the web. Content communication systems need a billing technique because such systems use money. However, a billing system isn't establish on unofficial sites. So, I introduce the SAS-Coin that we can use on mobile terminals. However, such technique has costs because it has high hash overhead. In this thesis, I research a new SAS-Coin system using the SAS-2 passward authentication system, and propose a Five-unit Coin system which is steady and has low costs.

key words SAS, SAS-Coin, mobile, billing protocol

目次

図目次

第 1 章

はじめに

1.1 背景

近年，インターネットの普及より，情報の多くは電子化・デジタル化され，多くの情報を素早く伝えることが可能となっている．インターネット上で音楽データや，画像データ，動画データといった，多数のコンテンツ流通が盛んに行われている．

最近では，モバイル端末によるインターネット接続率も増加し、モバイル端末上での情報交換も盛んに行なわれている．その中でも，携帯電話の爆発的な普及からモバイル端末の代表的存在となり，現在 7351 万 4100 台が普及している．インターネット利用可能な携帯端末においては，5952 万 7700 台に及ぶ．携帯電話からのインターネット接続の増加に伴い，モバイル端末上で有料・無料を問わずに音楽データや画像データなど多数のコンテンツ流通が盛んに行なわれるようになった．これから先もさらにコンテンツ流通が盛んに行われることが考えられる．

そこで，コンテンツ流通を円滑に進めていく一つの手法として，有料コンテンツの支払が必要であり，モバイル端末上での課金システムを確立することが重要な課題として挙げられる．

1.2 既存の技術

現在，携帯電話の有料コンテンツの料金支払い方法として，主に料金回収代行システムをとっている．料金代行者は i-mode では NTT ドコモ，ezweb では au，j-sky では J-phone

といったそれぞれのキャリアが提供サイトに替わって料金を回収している．料金回収代行システムは，携帯電話の公式サイトに登録されている場合にのみ適用されるサービスであり，公式サイトに登録されていない一万件以上ある勝手 (非公式) サイトでは，このシステムが適用されない．また公式サイトに入るためには厳しい審査があるために勝手サイトでは，他の会社が変わりに課金システムを提供している所や，カードを使った支払方法などを用いている．また公式サイトであっても，月額 300 円といったように定額料金で行われているところが多く，一つの情報量が 30 円といったように小額で払えるサイトはあまり多く無い，このような現状から複数回コンテンツを利用する人には使い勝手が良いが，一度や二度コンテンツを利用したいユーザにとって，定額料金でのコンテンツ利用は，好ましくない．

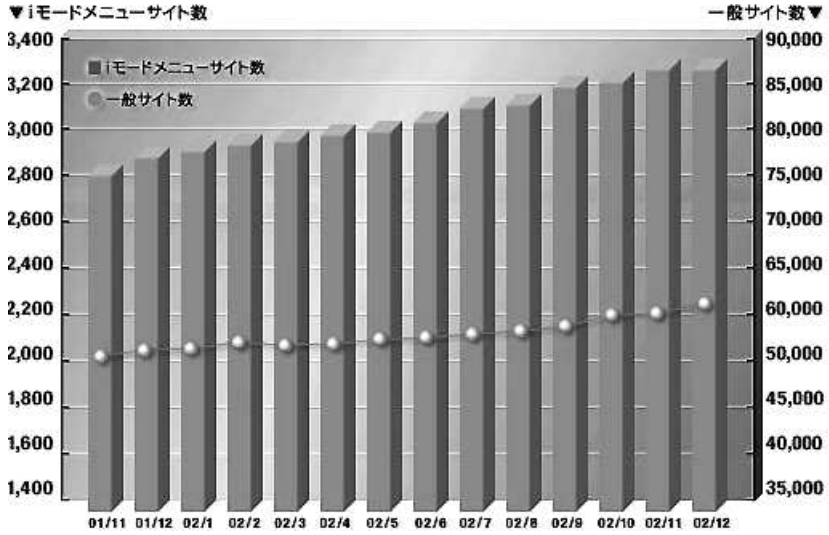

図 1.1　全体のサイト数と公式サイト数　([1] より引用)

1.3　目的

携帯電話上では定額料金でのコンテンツ流通が一般に行われているが，課金システムの確立ができたとは言いがたい．またどんなサイトにでも活用でき，ユーザにとって理解しやすい課金システムを作っていくことが，これから先必要となってくる．

そこで，小額課金システムの確立をすることにより，様々な有料コンテンツでの少数回利

用も可能となる．ユーザにとって少数回利用が可能になるということは，いろいろなサイトでの買い物も可能となり柔軟性が出てきて幅広い分野にも目を向けていくことができたため，インターネット上の買い物でも抵抗なく参加することができると考えられる．より日常に近い課金システムを確立していくことがこれから先のインターネットビジネスにも影響してくる．

小額課金システムの確立は，携帯電話の処理能力で実行できるシステムであり、安全性の高い技術が必要となってくる．現在様々な小額課金システムが提案されているが，本研究では SAS-Coin を取り上げる．小額課金システムの多くは公開鍵操作がネックとなっているが SAS-Coin では公開鍵を使わない方式であり，処理速度の低いモバイル端末に適用しやすい．また，他の小額課金システムの多くには備えていない，完全匿名である．携帯電話での使用を考えると処理速度の面からも問題が無く，匿名性を持っていることから他人にどこでお金を使用したかなどを知られないためにも有効である．

本研究では，研究室で提案された SAS(Simple And Secure password authentication protocol) ワンタイムパスワード認証方式 [2] を応用した小額支払システムである SAS-Coin の問題点を改良し，携帯電話での小額課金システムの構築を行う．さらに，支払い部分でのシミュレータを作成し，本システムの有効性を検討していく．

第 2 章

SAS-Coin

本節では，SAS 認証方式をベースにした SAS-Coin について記述する．本システムは Customer(ユーザ)，Vendor(コンテンツプロバイダ)，Broker(課金センタ) の三者で基本構成されている．Broker 側で，Vendor と Customer のアカウントを発行，保存，金銭の取引を行なう．また，初回の登録時のみ Agent(コンビニ等) が関与する．

2.1 表記記号の定義

本稿で用いる SAS-Coin についての表記記号の定義は以下のとおりである．

1. X : プリペイドカードに書かれているシリアル番号
2. Y : プリペイドカードにかかれているマスク処理された乱数
3. E^2_{nCB}=$E^2(S_C \oplus N_{nCB})$:Customer の Broker にたいする n 番目のデータ
4. E^2_{nCV}=$E^2(S_C \oplus N_{nCV})$:Customer の Vendor にたいする n 番目のデータ
5. E^2_{nVB}=$E^2(S_C \oplus N_{nVB})$:Vendor の Broker にたいする n 番目のデータ

2.2 SAS-Coin の手順

2.2.1 初期データ配分

初期データ配分には、プリペイドカードを用いた安全な方法を使用する．Broker は初期データ配分のためにカードを販売し．カードのシリアル番号Xをユーザ ID として使用する．

1. Broker : カードを製造し，コンビニエンスストア等に分配する．
2. Customer : カードを購入する．
3. Vendor : カードを入手する．
4. Customer : $X_C, E^2_{0CB} \oplus Y_C$ を Broker に送信する．
5. Broker : E^2_{nCB} を取得する．
6. Vendor : $X_V, E^2_{0VB} \oplus Y_V$ を Broker に送信する．
7. Broker : E^2_{nCB} を取得する．

2.2.2 コイン生成

カスタマはコインを生成し Broker，Vendor 側に課金を行なう．(図 2.1 参照)

1. Customer : 乱数 (RN) を生成，E^n_{RN} を計算し，以下のデータを Broker に送信する．
   - X_C, X_V : カスタマ ID とベンダ ID
   - $E^2_{0CB} \oplus E^n(RN)$，n : コインとコイン内の単位数 n
   - $E^3_{1CB} \oplus E^1_{0CB}$: 今回認証用
   - $E^2_{0CB} \oplus E^2_{1CB}$: 次回認証用
   - $E^2_{0CB} \oplus E^2_{0CV}$: ベンダ用認証データ
2. Broker : SAS を用いて認証し，次回認証用に，E^2_{0CB} を E^2_{1CB} に置き換える．そしてカスタマのアカウントに n 単位振込み，以下をベンダに送信する．
   - $E^2_{0VB} \oplus E^n(RN)$
   - $E^2_{0VB} \oplus E^2_{0CV}$，n$X_c$
3. Vendor : 認証後 $E^2(RN)$ と E^2_{0CV} を取得し，n，X_c をともに保存する．

2.2.3 支払いフェーズ

(図 2.2 参照)

1. Cusotmer : 以下をベンダに送信する.
   - X_C, $E^3_{1CV} \oplus E^1_{0CV}$: 今回認証用
   - $E^2_{0CV} \oplus E^2_{1CV}$: 次回認証用
   - $E^{n-t}(RN)$: t 単位支払い
2. Vendor : E^2_{0CV} を取得し，SAS を用いて認証し，次回認証用に E^2_{0CV} を E^2_{1CV} に置き換える．そして，$E^{n-t}(RN)$ に t 回ハッシュをかけ，$E^n(RN)$ でチェックする．正しければ情報を提供する.

2.2.4 回収フェーズ

(図 2.3 参照)

1. Vendor : 最終支払い ($E^{n-t}(RN)$ または RN) 受領後ブローカに提示する.
2. Broker : 最終支払い (n-t 回または n 回) にハッシュをかけ，ベンダのアカウントを検証し，そこから引き落とす．カスタマのアカウントが空きの場合，カスタマはお金を預ける際の ID して X を使い，匿名性を確保できる.

2.2 SAS-Coin の手順

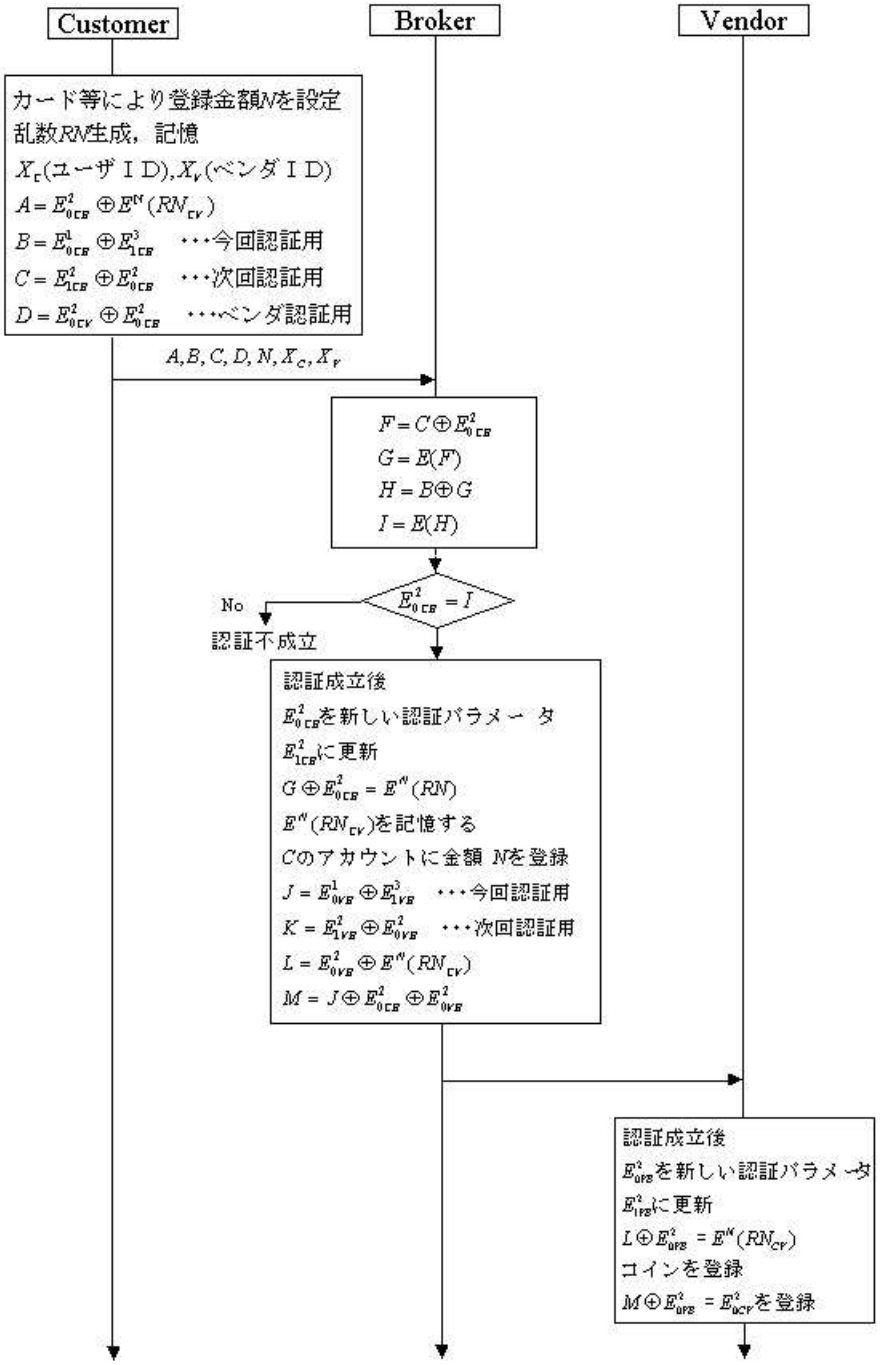

図 2.1　コイン生成

2.2　SAS-Coin の手順

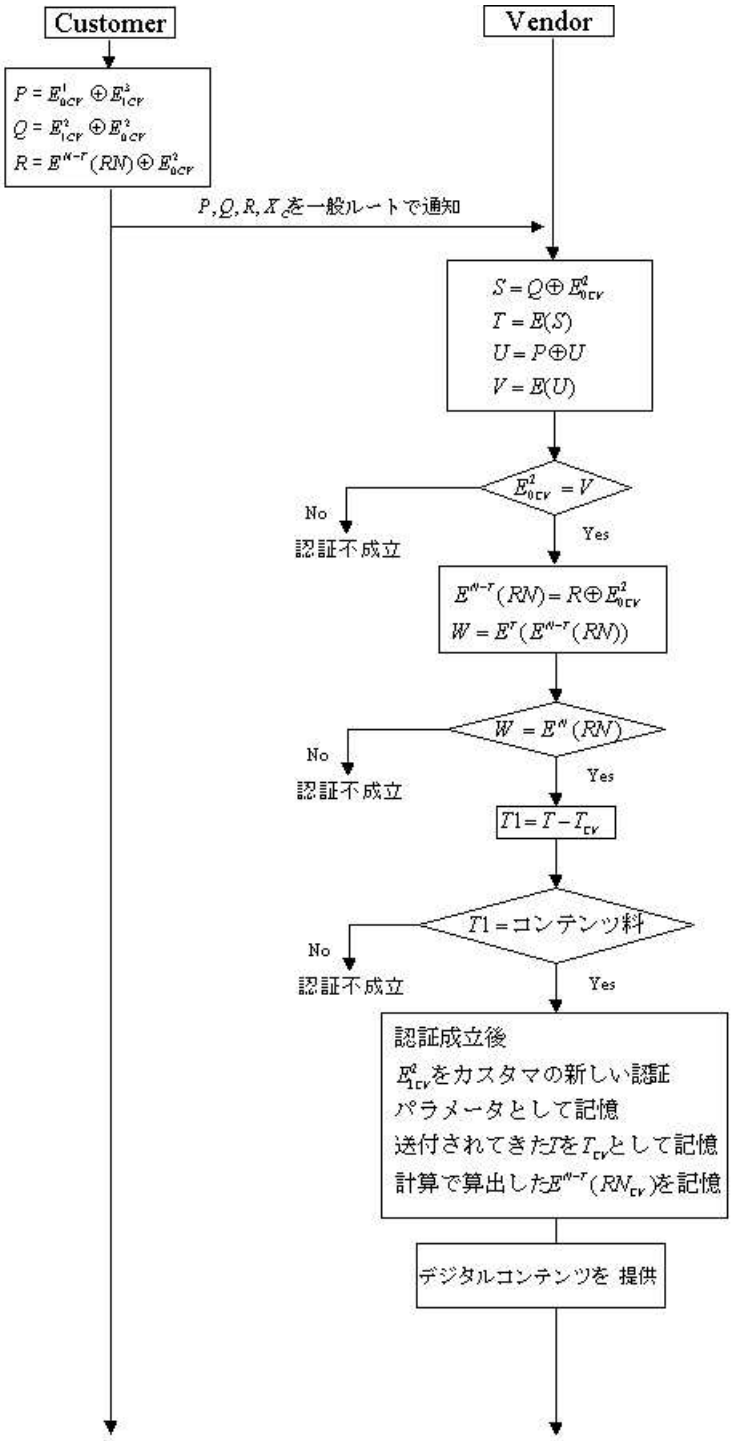

図 2.2　コイン生成

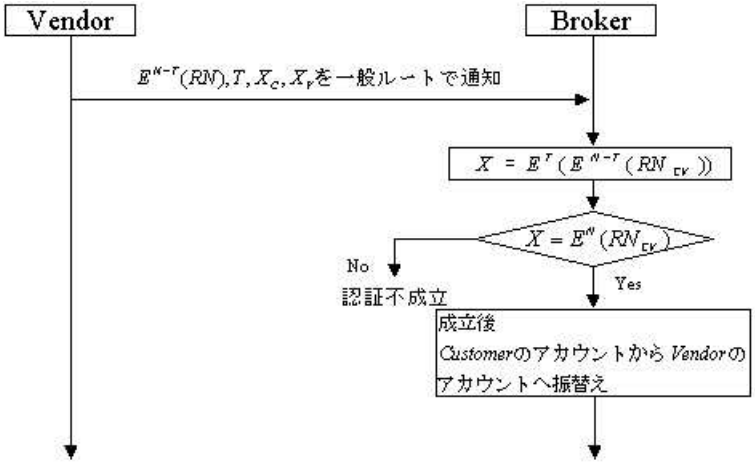
Vendor
Broker
$E^{N-T}(RN), T, X_C, X_P$を一般ルートで通知
$X = E^{T}(E^{N-T}(RN_{CV}))$
$X = E^{N}(RN_{CV})$
No
認証不成立
Yes
成立後
Customerのアカウントから Vendorの
アカウントへ振替え

図 2.3 回収フェーズ

第 3 章

SAS-Coin の問題点と解決法

現在，提案されている SAS-Coin を課金システムに当てはめるためには，様々な問題点が挙げられる．本章では，その問題点を取り上げ，その解決法まで記述する．

3.1 SAS-Coin の問題点

本研究室の実績から，現状で挙げられる問題点としては，以下のような問題点があげられる．

- 支払い時の計算量 (金額が大きくなると処理時間が必要)
- ベンダ固有

3.1.1 支払い時の計算量

SAS-Coin では，Customer が課金センタ (broker) 側に金額登録時，乱数に登録金額回ハッシュ関数を掛け，その値を送信し課金センタに登録する．支払い時はこの送られてきた．$E^{N-T}(RN)$ に支払い金額．T 回ハッシュ関数を乱数．RN にかけた値の差額を取りハッシュ演算を行い，Customer の課金金額と比較し同じ額なら支払いを行い完了する．この時、支払額が大きくなればなるほど計算量が増えることになる．$E^N(RN)$ の式で指数の N の値が大きくなるとき処理量も増加する．現在の流れでは，1000 円の支払いを行なうと 1000 回のハッシュ演算が必要となる．

3.1.2 ベンダ固有

SAS-Coin では，ベンダの変更ができないシステムだと言える．なぜなら，Customer 側から登録された認証情報は初回登録時に登録した Vendor にしか使用することができない．課金情報についても同様であり，他の Vendor を利用すると課金情報の同期が取れなくなる．他の Vendor を利用するときは新たに Broker 側に課金を行い，アカウントを発行してもらい，新たに認証情報を送らなければならない．同じコンテンツばかりを利用するときにはこのシステムでも良いが，新たなコンテンツを利用したい時に 1 コンテンツごとにアカウントが必要になるので多数のコンテンツを利用したいユーザにとっては使いづらい．

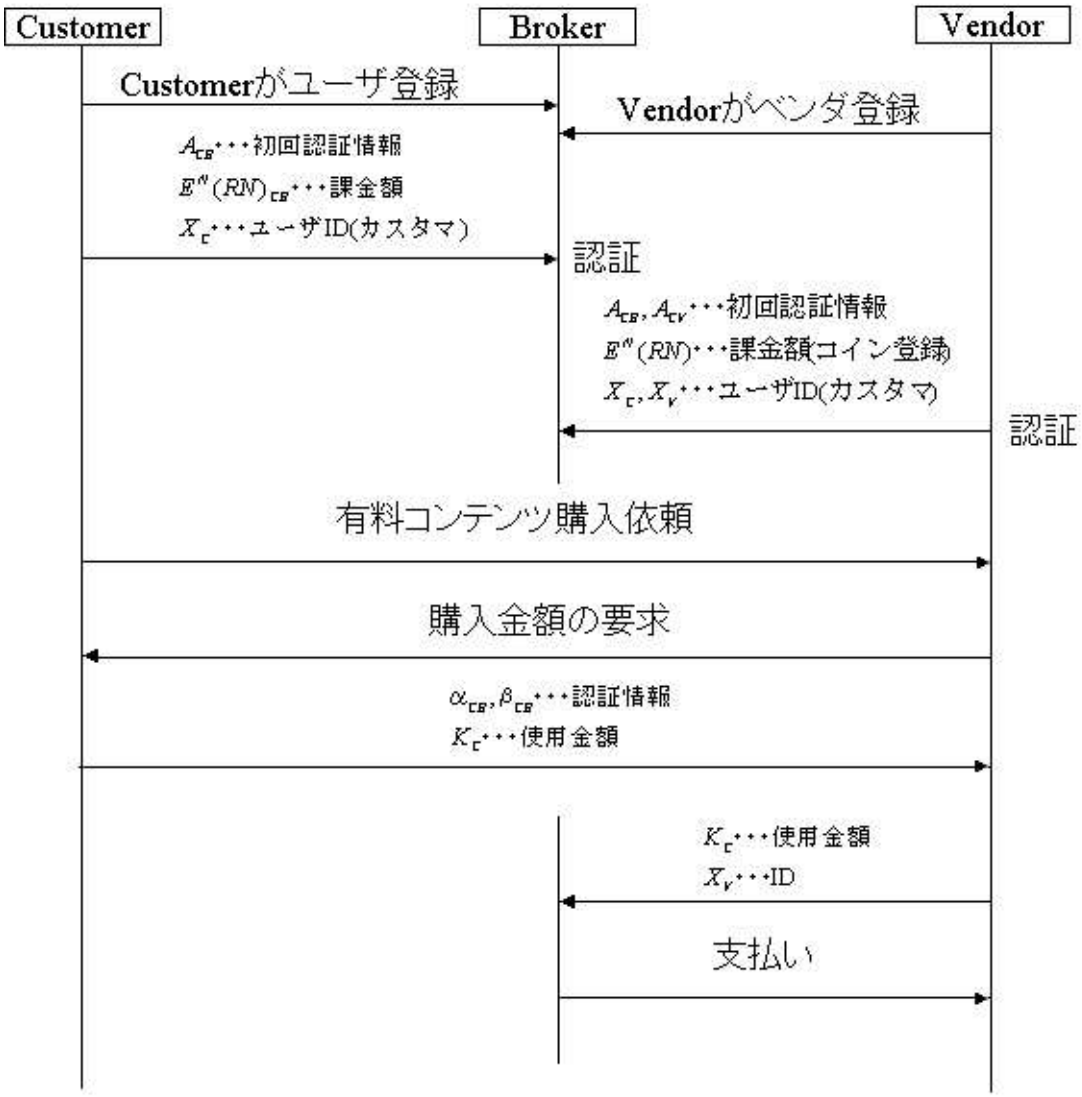

図 3.1　ベンダ固有

3.2　解決法

3.2.1　支払い時の計算量

処理量の解決法として，単位毎に決められた値を利用し，1 円，10 円，100 円，1000 円，10000 円単位ごとに計算することにより，計算量を減らすことができる．単位毎の計算を行なうために，それぞれの基本となる単位ごとの コインを作成する必要がある．提案した方

式では，単位ごとに計算するという方法をとるため，5 千円を支払う際に，5 千回ハッシュ演算する必要が無く，千円コインを 5 枚作成しこの値をを Vendor 側に送ればよい．この方式は，単位ごとにコインを設定する必要がある．以下の図でこの方式を示す.

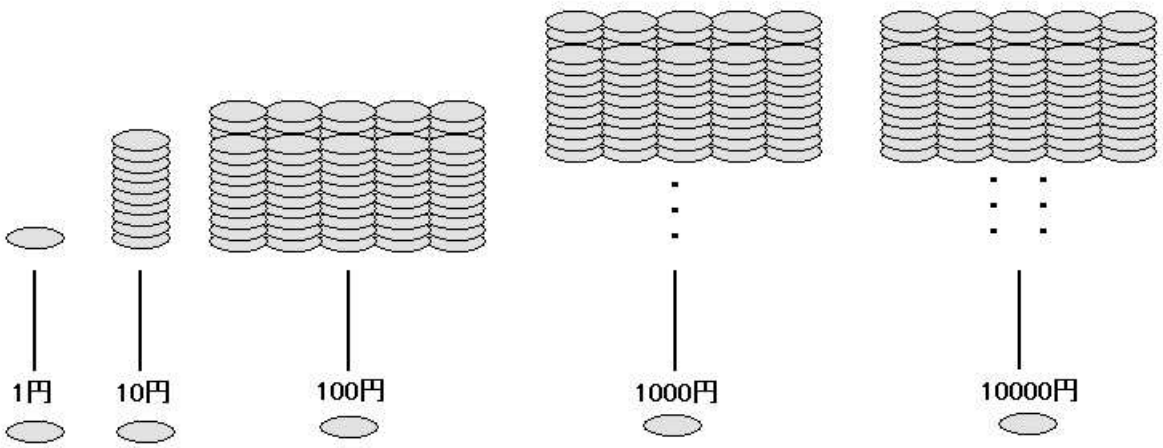

図 3.2　単位毎の計算

このようにあらかじめ基本となる単位を決めてやり単位毎仁演算することにより飛躍的に計算量が減る，SAS-Coin は小額課金方式でありさらに高額の支払いについてもセキュリティは十分である.

3.2.2　ベンダ固有

ベンダ固有の解決法として，SAS-Coin では CAA(コイン認証エージェント) を採用し解決を行っている [3]．そこで，本研究では，三者間通信のみでベンダの固有を解決する方法を提案する．これは，Vendor 側に課金情報を送らず支払い金額などを Broker 側で処理することにより Vendor を固定することなくシステムを構築することができる．以下に内容を示す.

図 3.3 では，Customer 側はベンタ固有の方式と変わらない流れになっている．そして Vendor 側では Customer-Vendor 間の認証情報，Customer の課金金額を保存させずに，Customer 側のコンテンツ購入情報をマスクして Broker 側に送る．Broker 側では，情報を

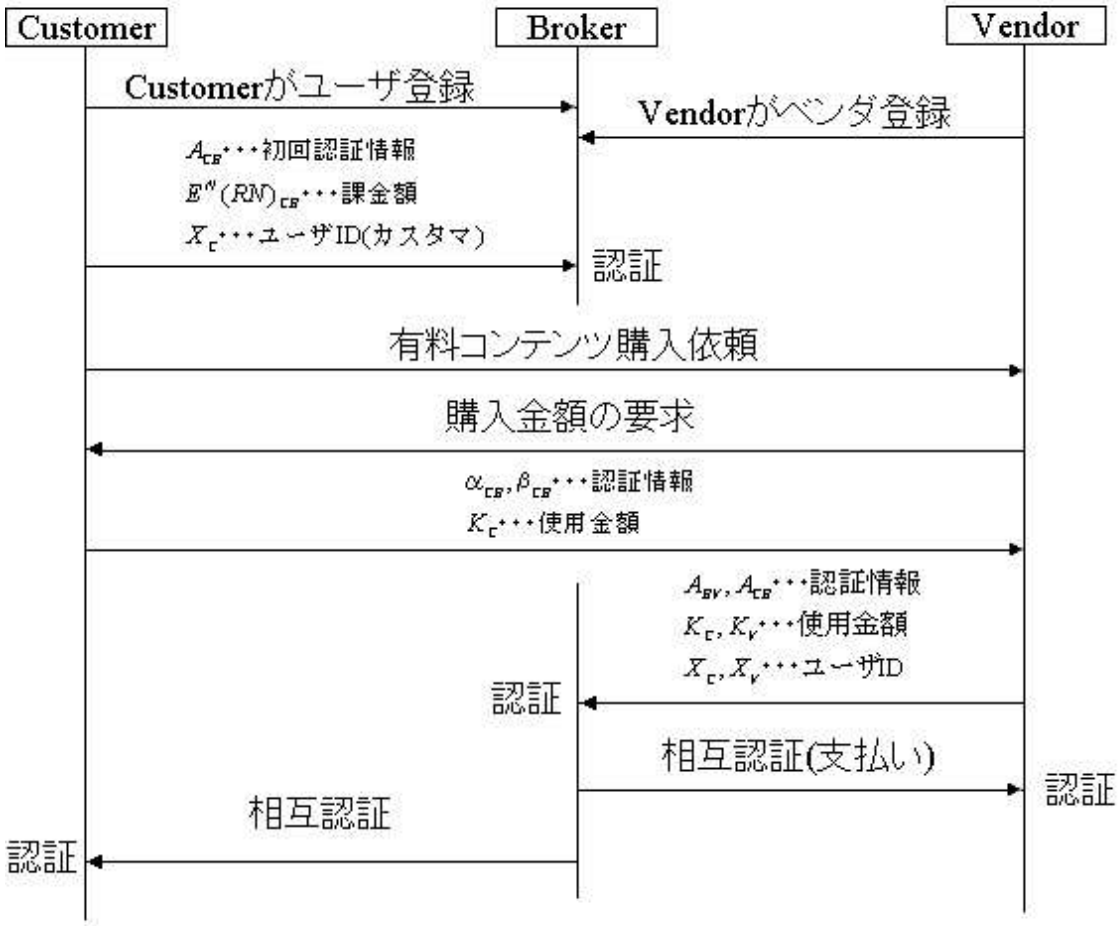

図 3.3　ベンダ固有の解決

入手し支払い，回収の行動に移す．それぞれの情報から Customer-Vendor 間で相互認証を行い，支払い，コンテンツの情報をそれぞれに送信する．

第 4 章

新たな方式の提案

本研究室で提案されている．SAS-2(Simple And Secure password authentication protocol, Ver.2) 認証方式 [4] を SAS-Coin に適用しより素早い処理の見込める小額課金システム SAS-2 Coin と SAS-2 認証方式のバリエーション [5] を利用した Five-unit Coin の構築をする．構築内容は以下に示す.

4.1 SAS-2 認証

新たに提案する SAS-2 Coin は SAS-2 ワンタイムパスワード認証方式を利用したシステムである．ここでは，SAS-2 認証による手順を説明する.

4.1.1 表記記号の定義

X:ID

S:Password

n:認証回数を示す 0 以上の整数

N_n:n 回目の認証で使用する乱数

A:N_n を用いて，一方向性関数を掛けたデータ (今回認証情報)

C:N_{n+1} を用いて，一 方向性関数を掛けたデータ (次回認証情報)

D:次回認証情報にハッシュ関数を掛けたデータ

三種類のハッシュ関数を使用,

E:ハッシュ関数 (一方向性関数)

F:ハッシュ関数

H:ハッシュ関数

α:マスク情報

β:マスク情報

⊕:排他的論理輪

←:出力

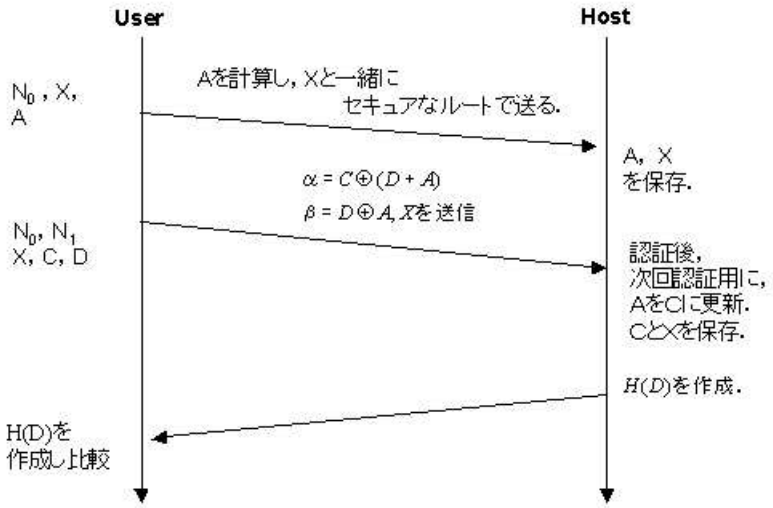

図 4.1　SAS-2

4.1.2　初回認証登録

1. ユーザは，Pasword と乱数 N_n を排他的論理輪で足したもの．

   A=E_0^1 ← E (ID，S ⊕ N_n) と ID=X を安全なルートでサーバに送信する．

2.　サーバでは，ユーザから送られてきた ID と初回認証情報である A=E_0^1 を保持しておく．

3.　ユーザは，Pasword と乱数 N_{n+1} から次回の認証情報である

   C=E_{n+1}^1← E (ID，S ⊕ N_{n+1}) を作成する．

4. 次回の認証情報である E^1_{n+1} にハッシュ関数 F を掛け

   D←F(E^1_{n+1}) を作成.

4.1.3 認証処理

1. ユーザは，サーバにサービス要求として

   α =C$\oplus$(D + A)

   β =D$\oplus$A

   ID

   の 3 点のデータを送信する.

2. サーバは，送られてきた情報に対し，事前に保持しておいたデータを用いユーザ認証を行う.

   ユーザから送られてきた，β に対し，所持しておいた A を排他的論理輪で足すことで D のデータを得られる.

   $\beta \oplus$A $\rightarrow$ D

   得られた D のデータを用い,

   $\alpha \oplus$(D + A) $\rightarrow$ C

   という計算を行うことで，次回認証情報である C を導きだすことができる.

   次に次回認証情報にハッシュ関数 F を掛ける計算を行う.

   F(E^1_{n+1})=D が得られる.

   D との比較を行ない認証の成立させする.

   D=D

3. 求められた D にハッシュ関数 H を掛け，H(D) を生成する.
4. 同様にユーザ側でも D というデータに対しハッシュ関数を掛け，H(D) を生成しておく.
5. サーバは H(D) というデータを安全なルートでユーザに送り，ユーザ側で生成しておいた H(D) の情報と比較する．データが一致することにより相互認証が成立する.

4.1.4 SAS-2 のバリエーション

SAS-2 には様々な認証パターンが存在する．本年度の研究成果によりセキュリティ面での安全性が確認されているバリエーションの中で 6 つの認証パターンを使用する．以下に各認証パターンを表記する．

- $\alpha \leftarrow$ A$\oplus$(F(C)+ID)

  $\beta \leftarrow$ F(C)$\oplus$C
- $\alpha \leftarrow$ F(C)$\oplus$(C+ID)

  $\beta \leftarrow$ F(C)$\oplus$A
- $\alpha \leftarrow$ C$\oplus$(F(C)+ID)

  $\beta \leftarrow$ F(C)$\oplus$A
- $\alpha \leftarrow$ C$\oplus$(F(C)+A)

  $\beta \leftarrow$ F(C)$\oplus$A
- $\alpha \leftarrow$ C$\oplus$(ID+A)

  $\beta \leftarrow$ F(C)$\oplus$A
- $\alpha \leftarrow$ A$\oplus$(ID+C)

  $\beta \leftarrow$ F(C)$\oplus$A

4.2 SAS-2 Coin

SAS-Coin に対して SAS-2 認証方式を適用する．SAS-Coin よりハッシュ計算が減少しているので少し速くなる．システム，流れに関しても SAS-Coin との違いはほとんど変わっていないが，認証部分での送信内容が SAS-2 に変更した物である．SAS 認証の場合 (図 4.2 参照) 今回認証情報で 2 回，次回認証情報で 3 回，合計 5 回のハッシュ演算が必要である．しかし SAS-2 認証 (図 4.1 参照) ではハッシュ演算が 3 回と抑えられているので，より軽く計算処理することが可能．このことからも，携帯電話の処理能力を考えると演算量が減ってい

るので，よりモバイル端末に適応できる方式であるといえる．

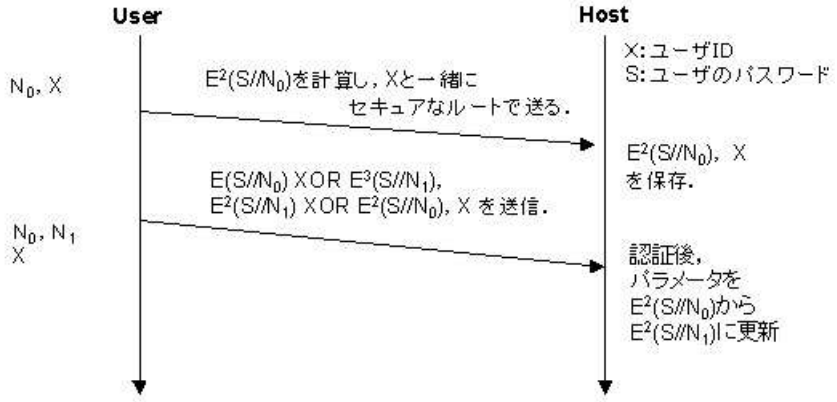

図 4.2 SAS 認証のフロー

4.3 Five-unit Coin の提案

4.3.1 Five-unit Coin の流れ

前節で提案した，SAS-2 認証を使用して作った SAS-2 Coin では，わずかな計算量の軽減は見込めた．しかし，SAS-2 Coin では支払い部分の処理量を削減するにいたらなかった．そこで，SAS-2 認証方式のバリエーションである認証方式を使用して計算量の少ない Five-unit Coin の提案を行う．

このモデルも Customer(ユーザ)，Broker(課金センター)，Vendor(コンテンツプロバイダ) の三者間で行う．また図 4.3 で Five-unit Coin の全体システム構成の小額課金プラットフォームについて記述する．

SAS-2 のバリエーションである認証方式中の 6 つのパターンをを利用し、それぞれの認証情報にコイン登録用，1，10，100，1000，10000 円という意味を持たせる事で単位毎の計算を行なう．

Five-unit Coin で使う認証情報とその意味

- $\alpha \leftarrow A \oplus (F(C)+ID)$ … 初回認証用

  $\beta \leftarrow F(C) \oplus C$

- $\alpha \leftarrow F(C) \oplus (C+ID)$ … 1 円コイン

  $\beta \leftarrow F(C) \oplus A$

- $\alpha \leftarrow C \oplus (F(C)+ID)$ … 10 円コイン

  $\beta \leftarrow F(C) \oplus A$

- $\alpha \leftarrow C \oplus (F(C)+A)$ … 100 円コイン

  $\beta \leftarrow F(C) \oplus A$

- $\alpha \leftarrow C \oplus (ID+A)$ … 1000 円コイン

  $\beta \leftarrow F(C) \oplus A$

- $\alpha \leftarrow A \oplus (ID+C)$ … 10000 円コイン

  $\beta \leftarrow F(C) \oplus A$

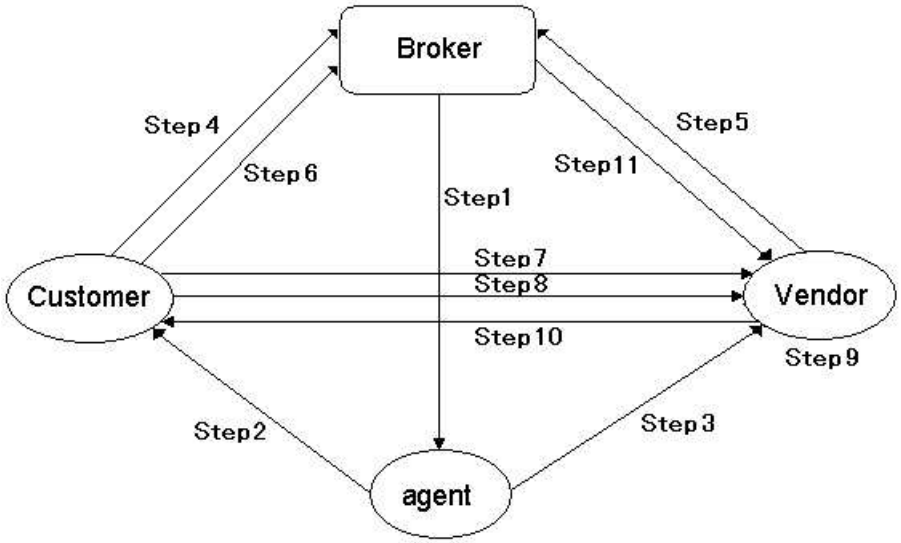

図 4.3 小額課金プラットフォーム

- Step1 カードを製造し Agent に分配する
- Step2 Agent からカード購入.

- Step3　Agent からカード購入.
- Step4　Customer がユーザ登録.（図 3.3)
- Step5　Vendor がベンダ登録.（図 3.4)
- Step6　Customer が金額登録を行う.（図 3.5)
- Step7　Customer がコンテンツを閲覧
- Step8　有料コンテンツ購入依頼.
- Step9　認証処理および課金.（図 3.6)
- Step10　デジタルコンテンツの送信.
- Step11　Vendor が支払い要求.（図 3.7)

Five-unit Coin 詳細なフローチャートを以下に示す．Five-unit Coin は SAS-Coin のプロトコルに対して SAS-2 プロトコルを適用することにより，より安全な小額課金システムの提案ができる．支払い時の計算量削減についても SAS-2 のバリエーションである 6 つの認証パターンを使い，1，10，100，1000，10000 ごとの計算を行えるようにし削減できる．

1. 図 4.4 は Customer の登録フローチャートを示す.（SAS-2 の登録手順)
   Agent からカードを購入，カードに書かれている情報と ID，初回の認証情報を生成したものを Broker 側に送信する．
2. 図 4.5 は Vendor 登録のフローチャートであり，Customer 登録と同手順で行う．
3. 図 4.6 金額登録のフローである．
   コインを生成し Customer，Vendor，Broker 間での情報の共有を行う．Customer は、Broker，Vendor，登録金額用に乱数をそれぞれ用意し，必要なデータを生成する必要がある．ID(カスタマ，ベンダ)，コインの情報，認証情報などを作成し，そのデータを Broker 側で認証し，金額を登録する，Broker 側は Customer-Vendor 間の認証情報と登録金額を Vendor 側に送り，Vendor 側はそれぞれを登録する．

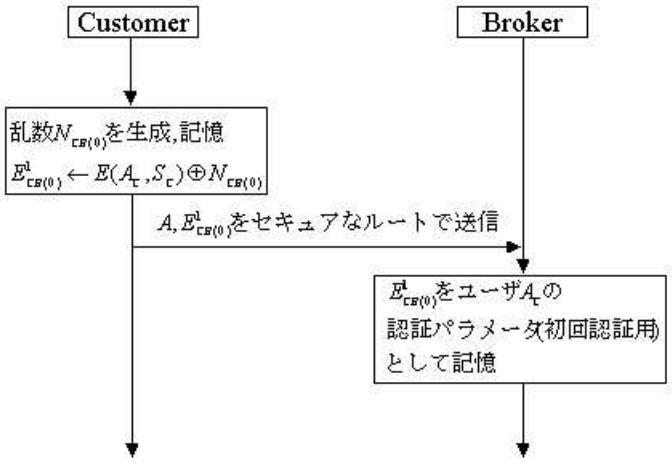

図 4.4 Customer 登録フロー

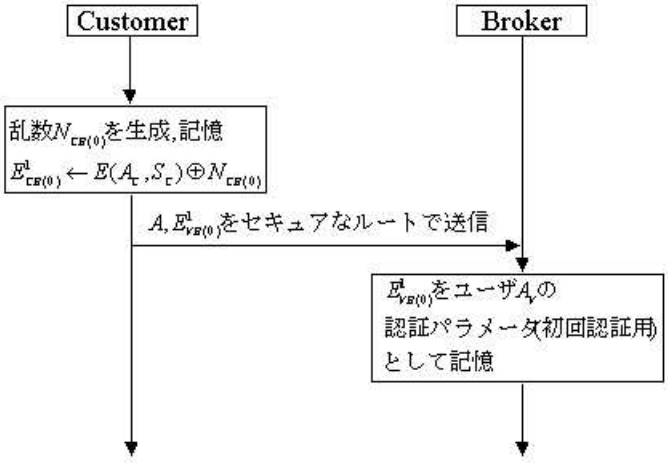

図 4.5 Vendor 登録フロー

4. 図 4.7 支払いフェーズのフローチャートである.

Customer は Vendor に認証情報とマスクしたコインの情報 (枚数)，使用金額 (今回を含む過去の使用金額の総計)，Customer の ID を送信する．Vendor 側では，認証情報、コインの情報，使用金額にたいしてそれぞれ認証を行っていく，始めに使用金額の確認を行う，次にコインの単位を確定させる，認証情報にコインの単位の情報も含まれているコインの単位の確認を行い，Customer の認証を行う．認証を行った後コインの情報

を数値化し支払いを行う.

5. 図 4.8 は回収フェーズのフローチャートである. Vendor 側は，Customer の利用金額とそれぞれの ID を Broker 側に送信，確認を行い Customer 側のアカウントから Vendor 側のアカウントに振込みを行う.

従来の SAS コインに SAS-2 を適用することにより，より速く，より安全性が優れた小額課金システムの構築を行った.

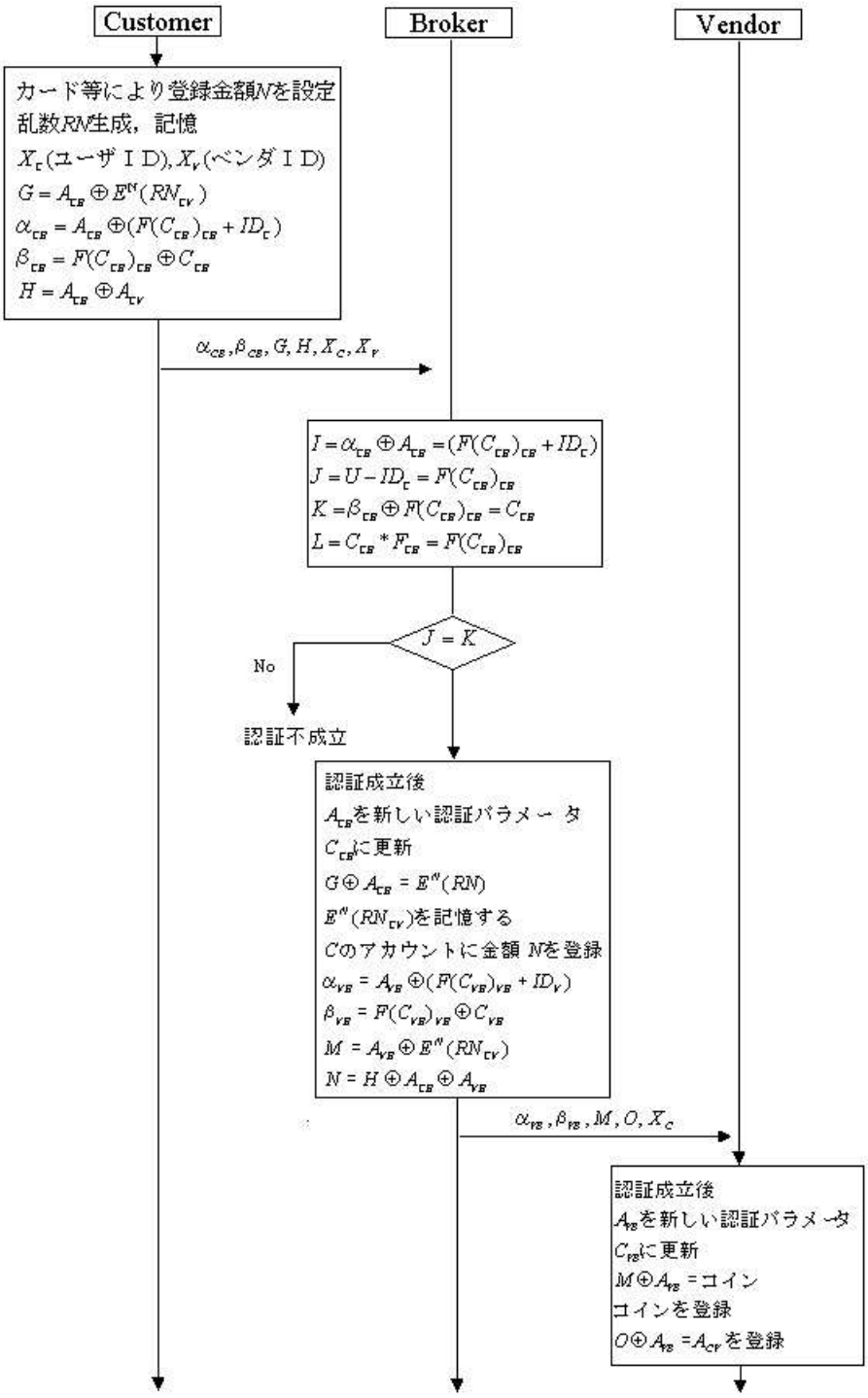
Customer
Broker
Vendor
カード等により登録金額Nを設定
乱数RNを生成，記憶
X_C(ユーザＩＤ), X_V(ベンダＩＤ)
$G = A_{CB} \oplus E^N(RN_{CV})$
$\alpha_{CB} = A_{CB} \oplus (F(C_{CB})_{CB} + ID_C)$
$\beta_{CB} = F(C_{CB})_{CB} \oplus C_{CB}$
$H = A_{CB} \oplus A_{CV}$
$\alpha_{CB}, \beta_{CB}, G, H, X_C, X_V$
$I = \alpha_{CB} \oplus A_{CB} = (F(C_{CB})_{CB} + ID_C)$
$J = U - ID_C = F(C_{CB})_{CB}$
$K = \beta_{CB} \oplus F(C_{CB})_{CB} = C_{CB}$
$L = C_{CB} * F_{CB} = F(C_{CB})_{CB}$
$J = K$
No
認証不成立
認証成立後
A_{CB}を新しい認証パラメータ
C_{CB}に更新
$G \oplus A_{CB} = E^n(RN)$
$E^n(RN_{CV})$を記憶する
Cのアカウントに金額 Nを登録
$\alpha_{VB} = A_{VB} \oplus (F(C_{VB})_{VB} + ID_V)$
$\beta_{VB} = F(C_{VB})_{VB} \oplus C_{VB}$
$M = A_{VB} \oplus E^n(RN_{CV})$
$N = H \oplus A_{CB} \oplus A_{VB}$
$\alpha_{VB}, \beta_{VB}, M, O, X_C$
認証成立後
A_{VB}を新しい認証パラメータ
C_{VB}に更新
$M \oplus A_{VB}$ =コイン
コインを登録
$O \oplus A_{VB} = A_{CV}$を登録

図 4.6　コイン生成

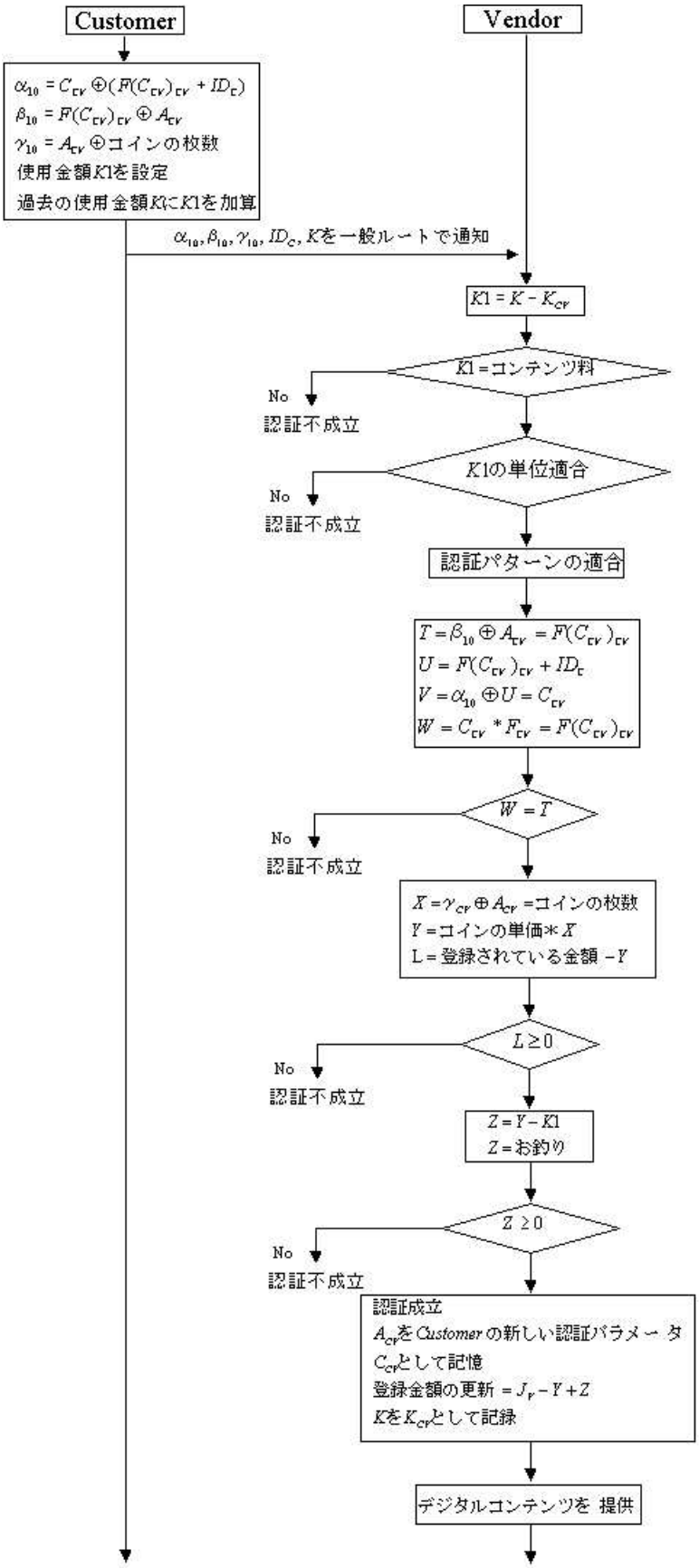
Customer
Vendor
$\alpha_{10} = C_{CV} \oplus (F(C_{CV})_{CV} + ID_C)$
$\beta_{10} = F(C_{CV})_{CV} \oplus A_{CV}$
$\gamma_{10} = A_{CV} \oplus$コインの枚数
使用金額K1を設定
過去の使用金額KにK1を加算
$\alpha_{10}, \beta_{10}, \gamma_{10}, ID_C, K$を一般ルートで通知
$K1 = K - K_{CV}$
K1=コンテンツ料
No
認証不成立
K1の単位適合
No
認証不成立
認証パターンの適合
$T = \beta_{10} \oplus A_{CV} = F(C_{CV})_{CV}$
$U = F(C_{CV})_{CV} + ID_C$
$V = \alpha_{10} \oplus U = C_{CV}$
$W = C_{CV} * F_{CV} = F(C_{CV})_{CV}$
$W = T$
No
認証不成立
$X = \gamma_{CV} \oplus A_{CV}$ =コインの枚数
Y =コインの単価$* X$
L=登録されている金額 $-Y$
$L \geq 0$
No
認証不成立
$Z = Y - K1$
Z =お釣り
$Z \geq 0$
No
認証不成立
認証成立
A_{CV}をCustomerの新しい認証パラメータ
C_{CV}として記憶
登録金額の更新 $= J_V - Y + Z$
KをK_{CV}として記録
デジタルコンテンツを 提供

図 4.7　支払いフェーズ

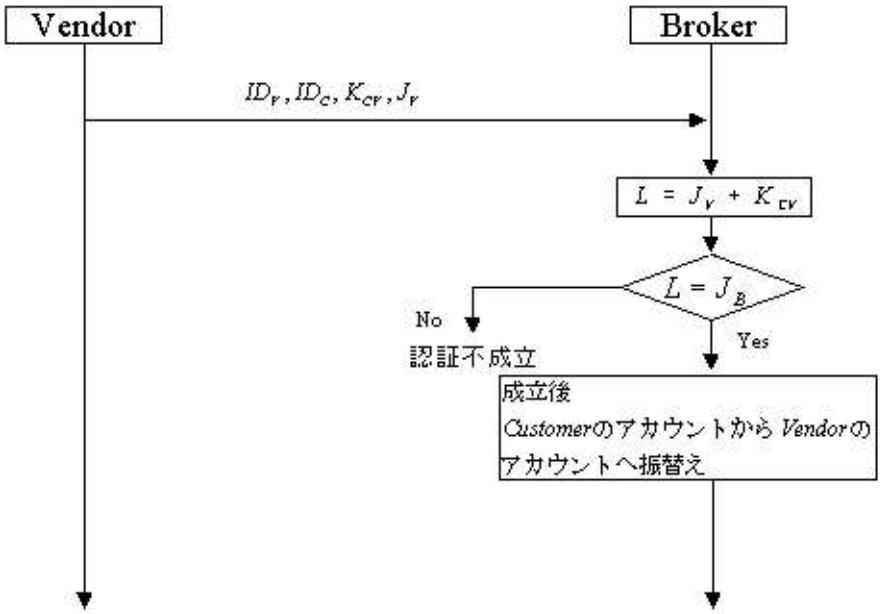
Vendor
Broker
ID_V, ID_C, K_{CV}, J_V
$L = J_V + K_{CV}$
$L = J_B$
No
認証不成立
Yes
成立後
Customerのアカウントから Vendorの
アカウントへ振替え

図 4.8 回収フェーズ

第 5 章

実装と評価

Five-unit Coin と SAS-2 Coin の支払いフェーズ部分での比較を行なう．評価内容としては，金額の違いにおける，処理時間，CPU 使用率を比較した．

5.1　シミュレータ

実験環境としては

- OS：Microsoft Windows 2000
- CPU：Celeron 1.7GHz
- メモリ:　256MB
- HDD：40G

のようにした．

4 章で提案した Five-unit Coin と SAS-Coin に SAS-2 認証を利用した SAS-2 Coin の支払い時間を計測し，どちらの方式が素早く支払いを終えることができるかの評価を行なった。Five-unit Coin の流れについては，第 4 章の図 4.6 参照，SAS-2 Coin については以下の図 5.1 にフローを記す．

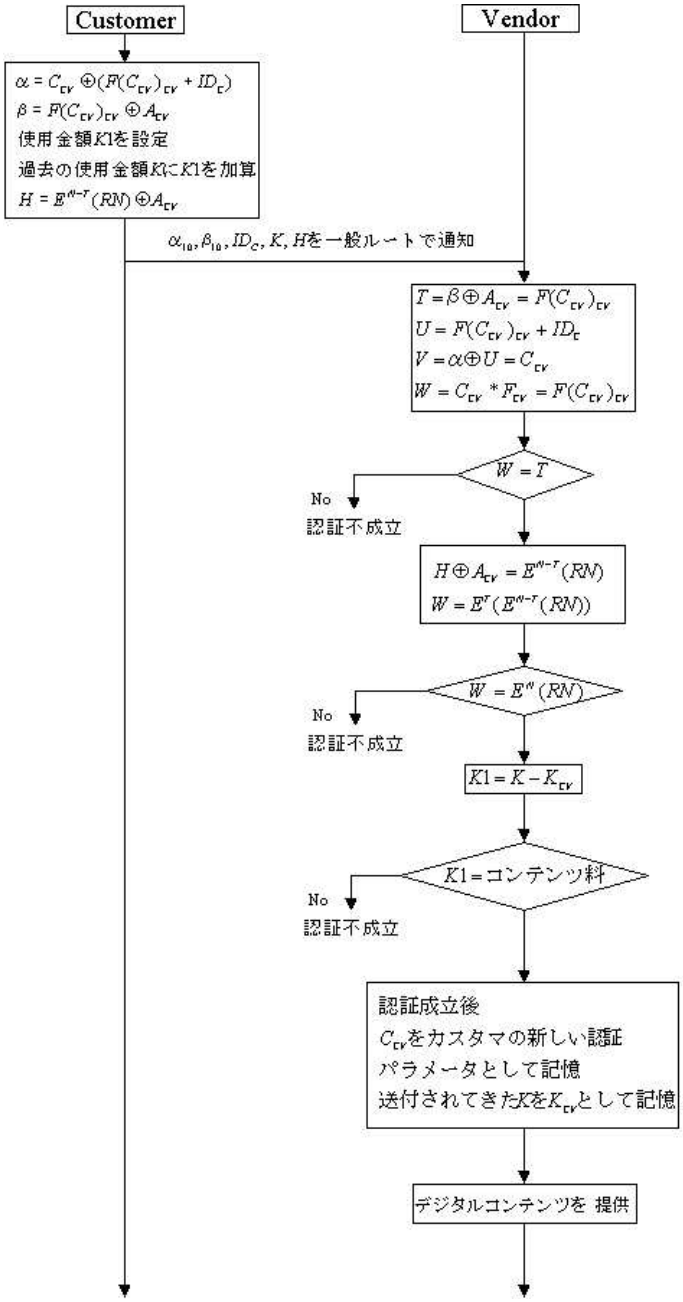

図 5.1　回収フェーズ

5.2　評価

評価の方法としては，以下の 2 点について評価を行なう．

- Customer 側が支払い金額を入力して，Vendor 側が認証から金額の支払い処理を終えるまでの時間を計測．

- Customer 側が支払い金額を入力して，Vendor 側が認証から金額の支払い処理を終えるまでの CPU 使用率を計測.

支払い金額の入力に関しては以下のパターンで進めた．各方式ともに課金金額は 100000 円で行い．1，2，3，4，5，6，7，8，9，10、20，30，40，50，60，70，80，90，100，200，300，400，500，600，700，800，900，1000，2000，3000，4000，5000，6000，7000，8000，9000，10000，20000，30000，40000，50000，60000，70000，80000，90000，100000 までそれぞれ支払い金額の入力を行い処理時間と CPU 使用率を計測した.

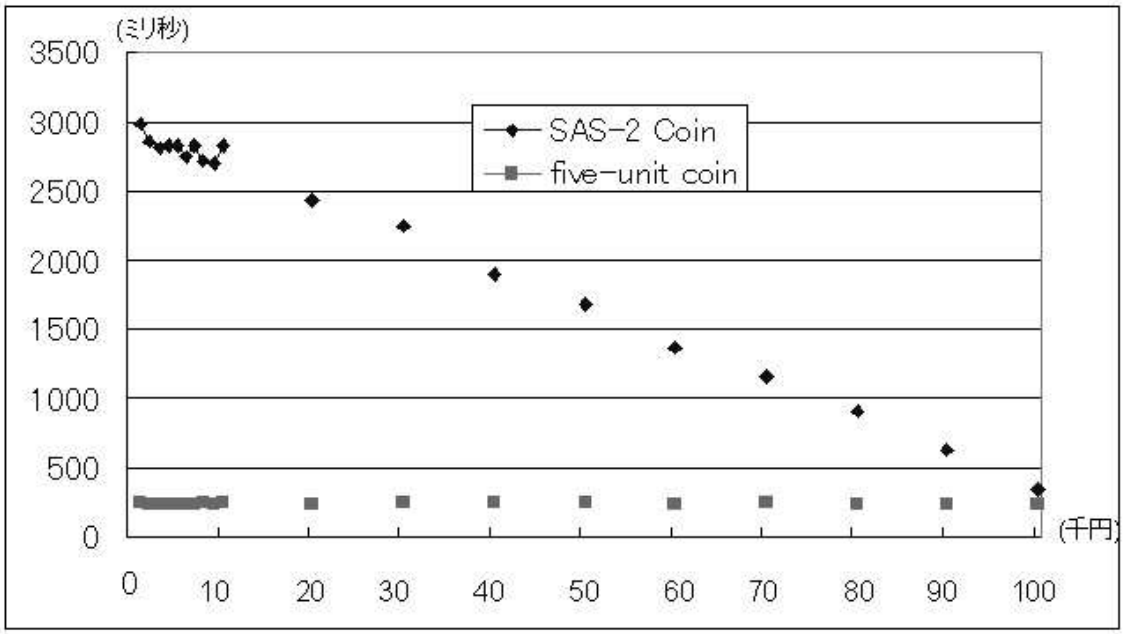

図 5.2　支払い時間比較

図 5.2 は各額の支払い時間の結果である.

- SAS-2 Coin は課金金額と支払額の差額をとって計算するので，課金金額に近づくほど処理時間が短くなる.
- Five-unit Coin はどんな支払額に対しても一定の時間で処理することが可能.

図 5.3 は各額の支払い時における CPU 使用率を計測した図である.

5.2　評価

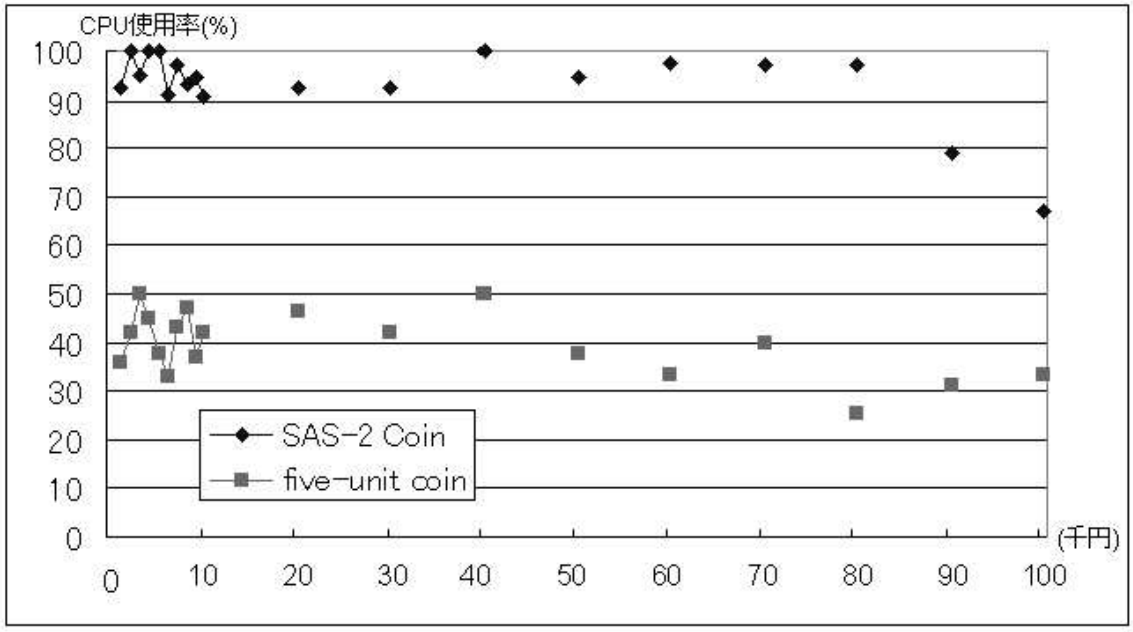

図 5.3　CPU 負荷率

- SAS-2 Coin の処理時は CPU 使用率は平均 90 %である,
- Five-unit Coin はほぼ SAS-2 Coin の CPU 使用率の半分で処理することが可能である.

モバイル端末に適応するときより処理負荷の低いシステムを利用していくことが必要である

上記の結果から

- Five-unit Coin は，支払額が大きくても小さくても処理時間がほぼ一緒である.
- 負荷の点においても Five-unit Coin は SAS-2 Coin よりも半分の負荷しかかからない.
- SAS-2 Coin は処理時間がかかるため処理負荷の高いこのシステムでは端末に対して負荷がかかりすぎる.
- Five-unit Coin は処理時間が少ないので端末にかかる負荷の割合が低い.

Five-unit Coin は SAS-2 Coin に比べて処理時間を大幅に削減し，CPU 使用率も SAS-2 Coin より半分の使用率で処理可能であった.

今後の課題としては,

- Customer 側，Vendor 側を各端末に振り分け実験を行ない負荷を調べる．
- コイン追加登録方法についての考察．
- コイン生成から回収までの実装を行い，処理時間と処理負荷の比較を行なう．
- おつりの支払いの時の処理時間・CPU 使用率

が挙げられる．

第 6 章

終わりに

本論文では，小額課金システムである，SAS-Coin の問題点について考察し，SAS-2 認証方式を SAS-Coin に応用するシステムを提案した．この方式では，単位毎の計算をする事により，支払い時の計算量の削減を可能とした．またベンダ固有の問題についても Vendor にコインの情報を保存させずに Broker 側で処理する事により Customer はいろいろな Vendor を利用する事が可能となった．この提案方式は，比較実験の結果からもわかるように，大幅に支払い時間の処理時間を短縮したシステムである．またモバイル端末へのを利用した実装実験をすることを今後の課題とする．

謝辞

高知工科大学工学部情報システム工学科　清水明宏教授には，研究室配属以来，就職活動，卒業研究を含め，学生生活全般に渡って懇切なるご指導，貴重な御教示を賜わった．ここに深謝申し上げる．

また，本学大学院生　工学研究科　基盤工学専攻　情報システム工学コース 2 回生　辻貴介氏，には，研究内容について有益な御議論をいただきご指導を賜った．本学大学院生　工学研究科　基盤工学専攻　情報システム工学コース　1 回生，大垣文誉氏，上岡隆氏，河村智氏，ならびに情報システム工学科 4 回生，池上奈津子氏，小西竜也氏，田岡慎也氏，永井慎太郎氏，奈良裕介氏，藤本卓氏，3 回生，上野真代氏，小川真依氏，福冨英次氏，には，実験準備から実験において御協力頂いた，ここに記して謝意を表する．

そして，在学中に御助言，ご指導くださった，諸先生方に心から感謝する．

参考文献

[1] DoCoMo Net

http://www.nttdocomo.co.jp/p_s/imode/

[2] M.Sandirigama, A. Shimizu, and M. Noda, “ Simple and Secure Password Authentication Protocol(SAS), ” IEICE Trans. Commun, vol.E83-B, no.6, pp.1363-1365, june 2000.

[3] M. Sandirigama, A. Shimizu, and M. Noda, “Simple and Secure Coin(SAS-Coin)-A Practical Micropayment System, ” IEICE Trans. Fundamentals, vol.E83-A, no.12, pp.2679-2688, December 2000.

[4] T. Tsuji, T. Kamioka, and A. Shimizu, “Simple and secure password authentication protocol, ver.2 (SAS-2), ”IEICE Technical Report, OIS2002-30, vol.102, no.314, pp.7-11, September 2002.

[5] T. Tsuji and A. Shimizu, “ Algorithm variations of SAS-2, ”IEICE Technical Report, IN2002-149, vol.102, no.498, pp.25-30, December 2002.